UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/804,735	03/19/2004	Li Long	INTEL/18495	8282

34431 7590 07/03/2007

HANLEY, FLIGHT & ZIMMERMAN, LLC
150 S. WACKER DRIVE
SUITE 2100
CHICAGO, IL 60606

EXAMINER	
WANG, BEN C	

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
07/03/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/804,735	LONG ET AL.	
	Examiner	Art Unit	
	Ben C. Wang	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *19 March 2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-11, 14, 16-18, 20-22, 24-27, and 30* is/are rejected.

7)☒ Claim(s) *12-13,15,19,23,28 and 29* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *07/16/2004*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-30 are pending in this application and presented for examination.

Claim Rejections – 35 USC § 101

35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

2. Claims 17-24 are rejected under 35 U.S.C 101 because the claims are directed to non-statutory subject matter.

3. **In claim 17**, an "instruction analysis module", a "cost estimation module", and a "partition generator", are being cited; however, it appears that the "instruction analysis module", the "cost estimation module", and the "partition generator" would reasonably be interpreted by one of ordinary skill in the art as computer listings per se, are not physical "things". They are neither computer components nor statutory processes, as they are not "act" being performed. Such claimed computer programs do not define any structural and functional interrelationships between the computer program and other claimed elements of a computer which permit the computer program's functionality to be realized. In contrast, a claimed computer readable medium encoded with a computer program is a computer element which defines structural and functional interrelationships between the computer program and the rest of the computer which permit the computer program's functionality to be realized, and is thus statutory. Accordingly, it is important

to distinguish claims that define descriptive material per se from claims that define statutory inventions. (See MPEP 2106.01(I))

4. **In claim 18,** a "redundant module" and a "mutual exclusion lock module" are being cited; however, it appears that the "redundant module" and the "mutual exclusion lock module" would reasonably be interpreted by one of ordinary skill in the art as computer listings per se, are not physical "things". They are neither computer components nor statutory processes, as they are not "act" being performed. Such claimed computer programs do not define any structural and functional interrelationships between the computer program and other claimed elements of a computer which permit the computer program's functionality to be realized. In contrast, a claimed computer readable medium encoded with a computer program is a computer element which defines structural and functional interrelationships between the computer program and the rest of the computer which permit the computer program's functionality to be realized, and is thus statutory. Accordingly, it is important to distinguish claims that define descriptive material per se from claims that define statutory inventions. (See MPEP 2106.01(I))

5. **As to claims** 21-24, they are merely further recited as the computer program product per se, thus, do not cure the deficiency of base claim 17, and also rejected under 35 U.S.C. 101 as set forth above.

6. **As to claims** 19-20, they are merely further recited as the computer program product per se, thus, do not cure the deficiency of base claims 17-18, and also rejected under 35 U.S.C. 101 as set forth above.

Claim Rejections – 35 USC § 112

The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

7. Claims 5 and 9 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

8. **Claim 5** recites the limitation "a virtual critical section" in line 2. There is not proper antecedent basis disclosed in the specifications. Examiner assumes “a virtual critical section” is “a critical section”.

9. **Claim 9** recites the limitation "a virtual critical section" in line 2. There is not proper antecedent basis disclosed in the specifications. Examiner assumes “a virtual critical section” is “a critical section”.

10. **Claim 19** recites the limitation "a second vector" in line 2. There is not proper antecedent basis disclosed in the specifications. Examiner assumes "a second vector" is "the second vector".

Claim Rejections – 35 USC § 102(b)

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102(b) that form the basis for the rejections under this section made in this office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

11. Claims 1-5, 8-9, 11, 16, 17, 22, 25-26, and 30 are rejected under 35 U.S.C. 102(b) as being anticipated by Tang et al. (*Thread Partitioning and Scheduling Based on Cost Model, 1997, ACM*) (hereinafter 'Tang')

12. **As to claim 1**, Tang discloses a method comprising: estimating a cost of merging a first set of instructions and a second set using a dataflow analysis (Abstract, 2nd Para., Lines 3-6 – based on a cost model, our algorithm groups instructions into thread by considering the trade-off among parallelism, latency tolerance, thread switching cost and sequential execution efficiency; Sec. 1.1 – An Example of Thread Partitioning, 1st Para., Lines 1-9, 13-14; 1.2 – Synopsis, 2nd Para., Lines 6-19; Sec. 2.2 – Program Modeling; Sec. 6 of Related Work, 2nd Para., Lines 1-3 – thread partitioning for multi-threaded execution models has been mainly done for functional languages

within the data-flow community; Reference, [4] – Integrating global caches and dataflow architecture, [5] – An evaluation of coarse-gain dataflow code generation strategies, and [13] – A dataflow/von Neumann hybrid architecture); and merging the first and second sets of instructions to form a merged set based on the cost of merging the first and second sets of instructions (Sec. 3 – Problem Statement, 8th Para., Lines 7-10 – two nodes can be merged into one thread to save thread switching costs; Sec. 4.1 – Overview of Thread Partitioning Heuristics, 4th Para., Lines 4-10 – We need to introduce the starting and finishing times of threads,....are merged into one thread; Sec. 4.2.1 – Thread Formation, 1st Para., Lines 2-6 – merging nodes reduces the number of threads generated and thus reduce the total coast due to thread switching; Sec. 5.5 – Thread Length, 4th Para., 4-6 - ...which merges a node into the same thread with one of its local predecessors, applies nearly 50% of time overall).

13. **As to claim 17**, Tang discloses an apparatus comprising: an instruction analysis module configured to perform a dataflow analysis (Sec. 4 – List-Scheduling Based Heuristic Algorithm; Sec. 6 – Related Work, 2nd Para.; Abstract, 2nd Para., Lines 3-6 – based on a cost model, our algorithm groups instructions into thread by considering the trade-off among parallelism, latency tolerance, thread switching cost and sequential execution efficiency; Sec. 1.1 – An Example of Thread Partitioning, 1st Para., Lines 1-9, 13-14; 1.2 – Synopsis, 2nd Para., Lines 6-19; Sec. 6 of Related Work, 2nd Para., Lines 1-3 – thread partitioning for multi-threaded execution models has been mainly done for functional languages within the data-flow community; Reference, [4] – Integrating global

caches and dataflow architecture, [5] – An evaluation of coarse-gain dataflow code generation strategies, and [13] – A dataflow/von Neumann hybrid architecture); a cost estimation module configured to determine an estimated cost of merging a first set of instructions and a second set of instructions to form a merged set of instructions (Sec. 2.2 – Program Modeling); and a partition generator (Sec. 1.1 – An Example of Thread Partitioning) configured to merge the first and second sets of instructions based on the estimated cost of merging the first and second sets of instructions (Sec. 3 – Problem Statement, 8th Para., Lines 7-10 – two nodes can be merged into one thread to save thread switching costs; Sec. 4.1 – Overview of Thread Partitioning Heuristics, 4th Para., Lines 4-10 – We need to introduce the starting and finishing times of threads,....are merged into one thread; Sec. 4.2.1 – Thread Formation, 1st Para., Lines 2-6 – merging nodes reduces the number of threads generated and thus reduce the total coast due to thread switching; Sec. 5.5 – Thread Length, 4th Para., 4-6 - ...which merges a node into the same thread with one of its local predecessors, applies nearly 50% of time overall).

14. **As to claim 25**, Tang discloses a machine readable medium having instructions stored thereon that, when executed, cause a machine to: estimate a cost of merging a first set of instructions and a second set of instructions using a dataflow analysis (Abstract, 2nd Para., Lines 3-6 – based on a cost model, our algorithm groups instructions into thread by considering the trade-off among parallelism, latency tolerance, thread switching cost and sequential execution efficiency; Sec. 1.1 – An Example of Thread Partitioning, 1st Para., Lines 1-9, 13-14; 1.2 – Synopsis, 2nd Para.,

Lines 6-19; Sec. 2.2 – Program Modeling; Sec. 6 of Related Work, 2nd Para., Lines 1-3 – thread partitioning for multi-threaded execution models has been mainly done for functional languages within the data-flow community; Reference, [4] – Integrating global caches and dataflow architecture, [5] – An evaluation of coarse-gain dataflow code generation strategies, and [13] – A dataflow/von Neumann hybrid architecture); and merge the first and the second sets of instructions to form a merged set of instructions based on the cost of merging the first and second sets of instructions (Sec. 3 – Problem Statement, 8th Para., Lines 7-10 – two nodes can be merged into one thread to save thread switching costs; Sec. 4.1 – Overview of Thread Partitioning Heuristics, 4th Para., Lines 4-10 – We need to introduce the starting and finishing times of threads,....are merged into one thread; Sec. 4.2.1 – Thread Formation, 1st Para., Lines 2-6 – merging nodes reduces the number of threads generated and thus reduce the total coast due to thread switching; Sec. 5.5 – Thread Length, 4th Para., 4-6 - ...which merges a node into the same thread with one of its local predecessors, applies nearly 50% of time overall).

15. **As to claim 2** (incorporating the rejection in claim 1), Tang discloses a method further comprising: estimating a cost of merging the first set of instructions and a third set of instructions; and estimating a cost of merging the second set of instructions and the third set of instructions (Sec. 2.2 – Program Modeling).

16. **As to claim 3** (incorporating the rejection in claim 2), Tang discloses a method wherein the cost of merging the first and third sets of instructions and the cost of

merging the second and third sets of instructions are greater than the cost of merging the first and second sets of instructions (Sec. 2.2 – Program Modeling).

17. **As to claim 4** (incorporating the rejection in claim 2), Tang discloses a method wherein the third set of instructions comprises a third critical section of instructions (Fig. 4 – An Annotated DDG and Its Optimum Partition; Sec. 3 – Problem Statement, 7th Para., Lines 1-4 – the difficulty in such scheduling is that the critical path of the partitioned threads is unknown …, 6-12; Sec. 4.1 – Overview of Thread Partitioning Heuristics, 1st Para., 1-7 – To help determine which nodes are critical,…., 3rd Para., Lines 3-10 – However, since h_4 is greater than both h_2 and h_3, we may ….on the critical path of final generated threads; Sec. 4.2.2 – Sequencing within Threads, 2nd Para., Lines 7-12 – However, v_j itself could be on the critical path,…).

18. **As to claim 5** (incorporating the rejection in claim 2), Tang discloses a method wherein the third set of instructions is associated with a virtual critical section (Fig. 4 – An Annotated DDG and Its Optimum Partition; Sec. 3 – Problem Statement, 7th Para., Lines 1-4 – the difficulty in such scheduling is that the critical path of the partitioned threads is unknown …, 6-12; Sec. 4.1 – Overview of Thread Partitioning Heuristics, 1st Para., 1-7 – To help determine which nodes are critical,…., 3rd Para., Lines 3-10 – However, since h_4 is greater than both h_2 and h_3, we may ….on the critical path of final generated threads; Sec. 4.2.2 – Sequencing within Threads, 2nd Para., Lines 7-12 – However, v_j itself could be on the critical path,…).

19. **As to claim 8** (incorporating the rejection in claim 1), Tang discloses a method wherein the first and second sets of instructions are associated with respective first and second critical sections (Fig. 4 – An Annotated DDG and Its Optimum Partition; Sec. 3 – Problem Statement, 7th Para., Lines 1-4 – the difficulty in such scheduling is that the critical path of the partitioned threads is unknown …, 6-12; Sec. 4.1 – Overview of Thread Partitioning Heuristics, 1st Para., 1-7 – To help determine which nodes are critical,…., 3rd Para., Lines 3-10 – However, since h_4 is greater than both h_2 and h_3, we may ….on the critical path of final generated threads; Sec. 4.2.2 – Sequencing within Threads, 2nd Para., Lines 7-12 – However, v_j itself could be on the critical path,…).

20. **As to claim 9** (incorporating the rejection in claim 1), Tang discloses a method wherein at least one of the first and second sets of instructions is associated with a virtual critical section (Fig. 4 – An Annotated DDG and Its Optimum Partition; Sec. 3 – Problem Statement, 7th Para., Lines 1-4 – the difficulty in such scheduling is that the critical path of the partitioned threads is unknown …, 6-12; Sec. 4.1 – Overview of Thread Partitioning Heuristics, 1st Para., 1-7 – To help determine which nodes are critical,…., 3rd Para., Lines 3-10 – However, since h_4 is greater than both h_2 and h_3, we may ….on the critical path of final generated threads; Sec. 4.2.2 – Sequencing within Threads, 2nd Para., Lines 7-12 – However, v_j itself could be on the critical path,…).

21. **As to claim 11** (incorporating the rejection in claim 1), Tang discloses a method wherein the cost of merging the first and second sets of instructions is associated with instructions that belong to only the first set of instructions and instructions that belong only to the second set of instructions Sec. 3 – Problem Statement, 8th Para., Lines 7-10 – two nodes can be merged into one thread to save thread switching costs; Sec. 4.1 – Overview of Thread Partitioning Heuristics, 4th Para., Lines 4-10 – We need to introduce the starting and finishing times of threads,....are merged into one thread; Sec. 4.2.1 – Thread Formation, 1st Para., Lines 2-6 – merging nodes reduces the number of threads generated and thus reduce the total coast due to thread switching; Sec. 5.5 – Thread Length, 4th Para., 4-6 - ...which merges a node into the same thread with one of its local predecessors, applies nearly 50% of time overall).

22. **As to claim 16** (incorporating the rejection in claim 1), Tang discloses a method further comprising creating a partition including the first and the second sets of instructions before the first and second sets of instructions are merged (i.e., Sec. 1.1 – An Example of Thread Partitioning, 1st Para., Lines 1-9, 13-14; 1.2 – Synopsis, 2nd Para., Lines 6-19; Sec. 2.2 – Program Modeling).

23. **As to claim 22** (incorporating the rejection in claim 17), Tang discloses an apparatus wherein the partition generator is configured to create a partition including the first and second sets of instructions before the first and second sets of instructions are

merged (i.e., Sec. 1.1 – An Example of Thread Partitioning, 1st Para., Lines 1-9, 13-14; 1.2 – Synopsis, 2nd Para., Lines 6-19; Sec. 2.2 – Program Modeling).

24. **As to claim 26** (incorporating the rejection in claim 25), Tang discloses a machine readable medium having instructions stored thereon that, when executed, cause the machine to: estimate a cost of merging the first set of instructions and a third set of instructions; and estimate a cost of merging the second set of instructions and the third set of instructions (Sec. 2.2 – Program Modeling).

25. **As to claim 30** (incorporating the rejection in claim 25), Tang discloses a machine readable medium having instructions stored thereon that, when executed, cause the machine to create a partition including the first and second sets of instructions before the first and second sets of instructions are merged (i.e., Sec. 1.1 – An Example of Thread Partitioning, 1st Para., Lines 1-9, 13-14; 1.2 – Synopsis, 2nd Para., Lines 6-19; Sec. 2.2 – Program Modeling).

Claim Rejections – 35 USC § 103(a)

26. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

27. Claims 6, 18, and 27 are rejected under 35 U.S.C. 103(a) as being unpatentable over Tang in view of Zoppetti et al., (*Automatic Compiler Techniques for Thread Coarsening for Multithreaded Architectures*) (hereinafter 'Zoppetti') and in further view of Buch et al., (Pub. No. US 2003/0065704 A1) (hereinafter 'Buch')

28. **As to claim 6** (incorporating the rejection in claim 1), Tang discloses thread partitioning based on cost model (Abstract, 2nd Para.) does not explicitly disclose a method further comprising: removing redundant instructions from the merged set of instructions; and assigning a physical mutual exclusion lock to the merged set of instructions.

However, in an analogous art of automatic compiler techniques for thread coarsening for multithreaded architectures, Zoppetti discloses a method further comprising: removing redundant instructions from the merged set of instructions (Sec. 1 – Introduction, 6th Para., 3rd bullet – use of must alias or definite points-to information for removing redundant remote references; Sec. 3.3 – Using Must Alias Information, 3rd Para., Lines 1-3; Sec. 6, 1st Para., Lines 6-9).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Zoppetti into the Tang's system to provide a method further comprising: removing redundant instructions from the merged set of instructions in Tang's system.

The motivation is that it would further enhance the Tang's system by advantageously taking, advancing and/or incorporating Zoppetti's system which

provides the techniques that lead to improve extraction and representation of dependence information in the presence of structured control flow....; The benefit of these techniques is the generation of coarser-gained threads and, therefore, decreased execution tome as once suggested by Zoppetti (i.e., Abstract; 2nd Para.).

Further, Tang discloses critical section of the partitioned threads (Sec. 3 – Problem Statement, 7th Para.), but both Tang and Zoppetti do not explicitly disclose assigning a physical mutual exclusion lock to the merged set of instructions.

However, in an art of flexible acceleration of Java™ thread synchronization on multiprocessor computers, Buch discloses assigning a physical mutual exclusion lock to the merged set of instructions ([0022], Lines 5-8 - ..a hardware lock has been assigned...).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Buch into the Tang-Zoppetti's system to further assign a physical mutual exclusion lock to the merged set of instructions in Tang-Zoppetti's system.

The motivation is that it would enhance the Tang-Zoppetti's system by taking, advancing and/or incorporating Buch's system which discloses that shared resources have associated "locks." A thread must acquire the lock on a resource in order to access the resource as once suggested by Buch (i.e., [0002], Lines 6-11).

29. **As to claim 18** (incorporating the rejection in claim 17), Tang discloses thread partitioning based on cost model (Abstract, 2nd Para.) does not explicitly disclose an

apparatus as defined further comprising: a redundant instruction module configured to remove redundant instructions from the merged set of instructions; and a mutual exclusion lock module configured to assign a first physical mutual exclusion lock to the merged set of instructions.

However, in an analogous art of automatic compiler techniques for thread coarsening for multithreaded architectures, Zoppetti discloses an apparatus as defined further comprising: a redundant instruction module configured to remove redundant instructions from the merged set of instructions (Sec. 1 – Introduction, 6th Para., 3rd bullet – use of must alias or definite points-to information for removing redundant remote references; Sec. 3.3 – Using Must Alias Information, 3rd Para., Lines 1-3; Sec. 6, 1st Para., Lines 6-9).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Zoppetti into the Tang's system to provide an apparatus as defined further comprising: a redundant instruction module configured to remove redundant instructions from the merged set of instructions in Tang's system.

The motivation is that it would further enhance the Tang's system by advantageously taking, advancing and/or incorporating Zoppetti's system which provides the techniques that lead to improve extraction and representation of dependence information in the presence of structured control flow....; The benefit of these techniques is the generation of coarser-gained threads and, therefore, decreased execution tome as once suggested by Zoppetti (i.e., Abstract; 2nd Para.).

Further, Tang discloses critical section of the partitioned threads (Sec. 3 – Problem Statement, 7th Para.), but both Tang and Zoppetti do not explicitly disclose a mutual exclusion lock module configured to assign a first physical mutual exclusion lock to the merged set of instructions.

However, in an art of flexible acceleration of Java™ thread synchronization on multiprocessor computers, Buch discloses a mutual exclusion lock module configured to assign a first physical mutual exclusion lock to the merged set of instructions ([0022], Lines 5-8 - ..a hardware lock has been assigned...).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Buch into the Tang-Zoppetti's system to further provide a mutual exclusion lock module configured to assign a first physical mutual exclusion lock to the merged set of instructions in Tang-Zoppetti's system.

The motivation is that it would enhance the Tang-Zoppetti's system by taking, advancing and/or incorporating Buch's system which discloses that shared resources have associated "locks." A thread must acquire the lock on a resource in order to access the resource as once suggested by Buch (i.e., [0002], Lines 6-11).

30. **As to claim 27** (incorporating the rejection in claim 25), Tang discloses thread partitioning based on cost model (Abstract, 2nd Para.) does not explicitly disclose a machine readable medium having instructions stored thereon that, when executed, cause the machine to: remove redundant instructions from the merged set of

instructions; and assign a physical mutual exclusion lock to the merged set of instructions.

However, in an analogous art of automatic compiler techniques for thread coarsening for multithreaded architectures, Zoppetti discloses a machine readable medium having instructions stored thereon that, when executed, cause the machine to: remove redundant instructions from the merged set of instructions (Sec. 1 – Introduction, 6th Para., 3rd bullet – use of must alias or definite points-to information for removing redundant remote references; Sec. 3.3 – Using Must Alias Information, 3rd Para., Lines 1-3; Sec. 6, 1st Para., Lines 6-9).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Zoppetti into the Tang's system to provide a machine readable medium having instructions stored thereon that, when executed, cause the machine to: remove redundant instructions from the merged set of instructions in Tang's system.

The motivation is that it would further enhance the Tang's system by advantageously taking, advancing and/or incorporating Zoppetti's system which provides the techniques that lead to improve extraction and representation of dependence information in the presence of structured control flow....; The benefit of these techniques is the generation of coarser-gained threads and, therefore, decreased execution tome as once suggested by Zoppetti (i.e., Abstract; 2nd Para.).

Further, Tang discloses critical section of the partitioned threads (Sec. 3 – Problem Statement, 7th Para.), but both Tang and Zoppetti do not explicitly disclose assigning a physical mutual exclusion lock to the merged set of instructions.

However, in an art of flexible acceleration of Java™ thread synchronization on multiprocessor computers, Buch discloses assigning a physical mutual exclusion lock to the merged set of instructions ([0022], Lines 5-8 - ..a hardware lock has been assigned...).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Buch into the Tang-Zoppetti's system to further assign a physical mutual exclusion lock to the merged set of instructions in Tang-Zoppetti's system.

The motivation is that it would enhance the Tang-Zoppetti's system by taking, advancing and/or incorporating Buch's system which discloses that shared resources have associated "locks." A thread must acquire the lock on a resource in order to access the resource as once suggested by Buch (i.e., [0002], Lines 6-11).

31. Claims 7 and 20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Tang in view of Zoppetti and Buch and in further view of T. Ogasawara et al., (Pat. No. US 7,089,540 B2) (hereinafter 'Ogasawara')

32. **As to claim 7** (incorporating the rejection in claim 6), Tang discloses thread partitioning based on cost model (Abstract, 2nd Para.) and Zoppetti discloses removing

redundant instructions (Sec. 1 – Introduction, 6th Para., 3rd bullet – use of must alias or definite points-to information for removing redundant remote references; Sec. 3.3 – Using Must Alias Information) but Tang, Zoppetti and Buch do not explicitly disclose a method wherein the redundant instructions comprise instructions used for at least one of entering a set of instructions and exiting the set of instructions.

However, in an analogous art of automatic compiler techniques for thread coarsening for multithreaded architectures, Ogasawara discloses a method wherein the redundant instructions comprise instructions used for at least one of entering a set of instructions and exiting the set of instructions (i.e., Fig. 2, element 140 – Synchronization Optimization Unit; Col. 3, Lines 30-32, 64-67; Col. 4, 36-39; Col. 9, Lines 16-24; Col. 12, Lines 48-50).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Ogasawara into the Tang-Zoppetti-Buch's system to provide a method wherein the redundant instructions comprise instructions used for at least one of entering a set of instructions and exiting the set of instructions in Tang-Zoppetti-Buch's system.

The motivation is that it would further enhance the Tang-Zoppetti-Buch's system by advantageously taking, advancing and/or incorporating Ogasawara's system which provides a synchronization optimization unit for deleting an unnecessary lock from the target program as once suggested by Ogasawara (i.e., Fig. 2, element 140 – Synchronization Optimization Unit; Col. 3, Lines 30-32, 64-67; Col. 4, 36-39).

33. **As to claim 20** (incorporating the rejection in claim 18), Tang discloses thread partitioning based on cost model (Abstract, 2nd Para.) and Zoppetti discloses removing redundant instructions (Sec. 1 – Introduction, 6th Para., 3rd bullet – use of must alias or definite points-to information for removing redundant remote references; Sec. 3.3 – Using Must Alias Information) but Tang, Zoppetti and Buch do not explicitly disclose an apparatus wherein the redundant instruction module is configured to remove redundant instructions comprising instructions for at least one of entering a set of instructions and exiting the set of instructions.

However, in an analogous art of automatic compiler techniques for thread coarsening for multithreaded architectures, Ogasawara discloses an apparatus wherein the redundant instruction module is configured to remove redundant instructions comprising instructions for at least one of entering a set of instructions and exiting the set of instructions (i.e., Fig. 2, element 140 – Synchronization Optimization Unit; Col. 3, Lines 30-32, 64-67; Col. 4, 36-39; Col. 9, Lines 16-24; Col. 12, Lines 48-50).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Ogasawara into the Tang-Zoppetti-Buch's system to provide an apparatus wherein the redundant instruction module is configured to remove redundant instructions comprising instructions for at least one of entering a set of instructions and exiting the set of instructions in Tang-Zoppetti-Buch's system.

The motivation is that it would further enhance the Tang-Zoppetti-Buch's system by advantageously taking, advancing and/or incorporating Ogasawara's system which

provides a synchronization optimization unit for deleting an unnecessary lock from the target program as once suggested by Ogasawara (i.e., Fig. 2, element 140 – Synchronization Optimization Unit; Col. 3, Lines 30-32, 64-67; Col. 4, 36-39).

34. Claims 10 and 21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Tang in view of Moon et al., (*Evaluation of Predicated Array Data-Flow Analysis for Automatic Parallelization*) (hereinafter 'Moon')

35. **As to claim 10** (incorporating the rejection in claim 1), Tang discloses using a dataflow analysis (Sec. 6 of Related Work, 2nd Para.) but does not explicitly disclose a method wherein the dataflow analysis comprises a forward disjunctive dataflow analysis.

However, in an analogous art of automatic compiler techniques for thread coarsening for multithreaded architectures, Moon discloses a method wherein the dataflow analysis comprises a forward disjunctive dataflow analysis (Sec. 4.1 – Predicate Domain, 4th Para., Lines 1-4, 5th Para., Lines 1-5).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Moon into the Tang's system to provide a method wherein the dataflow analysis comprises a forward disjunctive dataflow analysis in Tang's system.

The motivation is that it would enhance the Tang's system by taking, advancing and/or incorporating Moon's system which provides two distinguished features (1) it

derives low-cost, run-time parallelization test; and, (2) it incorporates predicate embedding and predicate extraction, which translate between the domain of predicates and data-flow values to derive more precise analysis results as once suggested by Moon (i.e., Abstract; 2nd Para., Lines 2-7).

36. **As to claim 21** (incorporating the rejection in claim 17), Tang discloses using a dataflow analysis (Sec. 6 of Related Work, 2nd Para.) but does not explicitly disclose an apparatus wherein the instruction analysis module is configured to perform a forward disjunctive dataflow analysis.

However, in an analogous art of automatic compiler techniques for thread coarsening for multithreaded architectures, Moon discloses an apparatus wherein the instruction analysis module is configured to perform a forward disjunctive dataflow analysis (Sec. 4.1 – Predicate Domain, 4th Para., Lines 1-4, 5th Para., Lines 1-5).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Moon into the Tang's system to provide an apparatus wherein the instruction analysis module is configured to perform a forward disjunctive dataflow analysis in Tang's system.

The motivation is that it would enhance the Tang's system by taking, advancing and/or incorporating Moon's system which provides two distinguished features (1) it derives low-cost, run-time parallelization test; and, (2) it incorporates predicate embedding and predicate extraction, which translate between the domain of predicates

and data-flow values to derive more precise analysis results as once suggested by Moon (i.e., Abstract; 2nd Para., Lines 2-7).

Allowable Subject Matter

37. Claims 12-15 and 28-29 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten to overcome all the limitations of the base claim and any intervening claims.

The following is an examiner's statement of reasons for allowance:

38. **Regarding claims** 12-15 and 28-29, prior art of record fails to reasonably show or suggest the specific vectors created based on the dataflow analysis, wherein elements of the first vector comprise instructions contained in at least one of the first and second sets of instructions and the cost matrix created based on the first vector, wherein the cost matrix contains the cost of merging the first and second sets of instructions as claimed. Further, the second vector comprising a redundancy indicator after merging the first and the second set of instructions, furthermore updating the first vector and the cost matrix after merging the first and second sets of instructions.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Conclusion

39. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

- Franssen et al., Control Flow and Memory Management Optimization (Pat. No. 6,064,819)
- Schauser et al., Separation Constraint Partitioning – A New Algorithm for Partitioning Non-strict Programs into Sequential Threads, 1995, ACM
- Tang et al., How "hard" Is Thread Partitioning and How "bad" Is a List Scheduling Based Partitioning Algorithm, 1998, ACM
- T. J. Biggerstaff et al., Anticipatory Optimization with Composite Folding (Pat. No. 6,745,384 B1)

40. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is 571-270-1240. The examiner can normally be reached on Monday - Friday, 8:00 a.m. - 5:00 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

TUAN DAM
SUPERVISORY PATENT EXAMINER

BCW BW

June 25, 2007